A Study on Localization of Knowledge in Language Models

Steve Wen*
Student ID 260948939
McGill / Computer Science
shuo.wen@mail.mcgill.ca

Mark Bai*
Student ID 260788233
McGill/MILA
hao.bai@mail.mcgill.ca

Zichao Li
Mentor
McGill/MILA

Abstract

With the growing popularity of large language models, it is essential to develop efficient and effective techniques to update obsolete knowledge within the model without re-training on extremely large datasets. To do so, it is essential to first locate knowledge in the models. Some have approached this problem with gradient attribution, while others have used network analysis to selectively edit neurons in critical information pathways. To evaluate the result of these localization techniques, we applied them to smaller language models, and found that knowledge is stored in the upper MLP layers in GPT-2. We then applied 2 editing techniques on the 2 different proposed locations of knowledge, and evaluated their reliability, generality, and locality.

1 Introduction

Large language models (LLM) based on Transformer architecture have harnessed great power in various natural language-related tasks (Vaswani et al., 2017; Devlin et al., 2019; Brown et al., 2020). Those models have achieved near or even surpassed human-level performances in certain aspects (He et al., 2021; Raffel et al., 2020). Previous works also showed that implicit knowledge is stored within the architectures of pre-trained LLMs (Petroni et al., 2019; Jiang et al., 2020; Elazar et al., 2021a). While some have focused on masked BERT (Bidirectional Encoder Representations from Transformers) models (Devlin et al., 2018), autoregressive GPT (Generative Pre-trained Transformer) models (Radford et al., 2018, 2019) are especially interesting due to their unidirectional attention and generative properties. However, a remaining main challenge here is how to rapidly modify the localized factual knowledge of the pre-trained model to make it up-to-date with the constantly changing world. For example, a model trained before 2020 prompted with the question: *Who is the president of the US?* would assign a higher probability to *Trump* than to *Joe Biden*, which would generate misleading answers if the model is deployed after 2020. Updating the whole model knowledge is both computationally and data-expensive, so we would want to modify this specific knowledge.

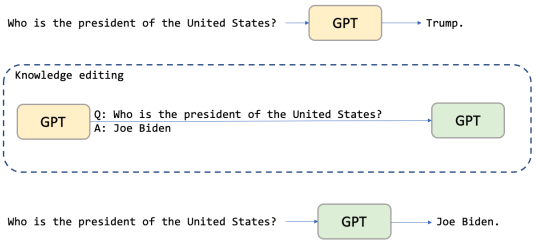


Figure 1: **An illustration of Model Knowledge Editing**

Ideally, when the model has outdated knowledge, we want to make a *swift* update to the answers of the prompted question **(reliability)**, as well as its related questions **(generality)**, without affecting unrelated stored knowledge **(locality)**. In recent years, fine-tuning with constraint (Zhu et al., 2020), meta-learning approaches with hypernetworks (Mitchell et al., 2022a; De Cao et al., 2021) have shown promising results. More than that, constrained least-square optimization provided a novel insight into substructure editing and reconfirmed that middle MLP (multilayer perceptron) layers of the LLM are crucial to storing knowledge (Meng et al., 2022b,a; Geva et al., 2021). All these works are dedicated to pursuing high efficacy while retaining efficient computation. While *reliability* and *generality* are maintained, its *locality* is often compromised for the others. We hypothesized this largely accredits to insufficient conditions are actually implemented to control the stability of the edit, and they did not make full use of the storage effects of the middle layers.

Based on Meng et al. (2022b,a); Geva et al.

*These authors contributed equally to this work

(2021), we knew certain layers have dominant indirect effects over others. Intuitively, carefully designed weight updates should be assigned to these modules, which is in accordance with the work by Dai et al. (2022). However, previous works did not have special treatments on these layers. While there is an agreement that the MLP layers of LLMs are important in knowledge retrieval, it is unclear which layers are the most crucial. For instance, Meng et al. (2022a) believe the middle MLP layers are the most important. Inspired by Vig et al. (2020); Pearl (2022)'s neuron intervention technique on language models, we hypothesize different neurons in that layer should have respective causal effects. Naturally, neurons with high causal association to factual knowledge storage should be treated with optimized weight updates. However, with the emergence of prompt learning paradigms (Wang et al., 2022; Lester et al., 2021), it is also possible that knowledge is stored in the prompts of the knowledge probing tasks, instead of the neurons. In this paper, we first extend the work of Meng et al. (2022a) and Dai et al. (2022) on smaller language models (BERT by Devlin et al. (2018), and GPT-2 and GPT-2 medium by Radford et al. (2019)). We then investigate the properties of syntactic knowledge neurons while examining the importance of prompts in knowledge retrieval. Finally, to investigate the disagreement on the location of MLP layer knowledge storage, we then apply knowledge neuron editing techniques on the causal neurons that are identified by causal mediation analysis, and apply ROME (Rank-One Model Editing) on knowledge neurons identified in knowledge neurons. This part of the experiment is only carried out on autoregressive GPT models.

2 Related Works

2.1 Knowledge in Language Models

Since LLMs can answer natural-language queries about real-world facts, it has been proposed that they could be used directly as knowledge bases (Petroni et al., 2019). Geva et al. (2021) identify the MLP layers in an autoregressive transformer as key–value memories. Building on this and Vig et al. (2020)'s work on analyzing gender bias in LLM components, (Meng et al., 2022a) have used causal mediation analysis to trace the causal effects of hidden state activations within GPT. Causal mediation analysis aims to measure how something's effect is mediated by intermediate variables. For instance, drug A's side effect may cause someone to take drug B, which has a separate positive/negative effect on the disease from drug A. Similarly, in a neural network, neurons are influenced by the input, and then affect the model output; however, there also exist direct pathways from the input to the output that do not pass through some neurons. Another branch of work by (Dai et al., 2022; Hao et al., 2021) uses gradient integration to evaluate the contribution of neurons to knowledge retrieval. While both studies agree on the idea that MLPs in transformers serve as key-value pair memories (Geva et al., 2021), they disagree on the exact MLP layers. Specifically, Meng et al. (2022a) believe that mid-layer MLPs output properties about a subject, which are then copied to the embedding to the last token by attention at the higher attention layers in transformers. On the other hand, (Dai et al., 2022) find knowledge neurons that are positively correlated to knowledge expression. Importantly, these knowledge neurons are found to be concentrated in the last few MLP layers and do not necessarily correspond to the last token of the subject in knowledge tuples. Surprisingly, they also showed promising results in knowledge editing by simplifying adding the word embedding of the new fact to the knowledge neuron and similarly subtracting the word embedding of the old fact.

2.2 Knowledge Editing

The goal of knowledge editing in language models is the following: given a model $f_\theta : \mathcal{X} \times \Theta \rightarrow \mathcal{Y}$ that maps an input x and a set of parameters θ to an output y, how can we use a single pair of input-output (x_e, y_e) such that the output of $f_{\tilde{\theta}}$ ($\tilde{\theta}$ is the updated set of parameters) is altered on all semantically equivalent input-output pairs of (x_e, y_e). The most straightforward strategy would be to re-train the model on a new dataset with equivalent pairs of (x_e, y_e) included. However, given the rise of LLMs, editing by re-training is currently expensive and infeasible. Some have tried fine-tuning (Sotoudeh and Thakur, 2019; Zhu et al., 2020), but this is not only potentially expensive but also leads to overfitting which compromises the locality and generality of the update.

Hypernetwork model editors utilize a meta-learning approach (Ha et al., 2017) to learn to predict the updated $\tilde{\theta}$. For instance, Knowledge Editor (KE) trains a recurrent neural network to map the edit example into a rank-1 mask over the gradient

(De Cao et al., 2021). In Model Editor Networks with Gradient Decomposition (MEND), however, the fine-tuning gradient at layer l is directly mapped to parameter edits of layer l (Mitchell et al., 2022a). This is done with a collection of trainable small auxiliary networks that receive rank-1 matrices containing batch gradients, splits them into corresponding vectors, and output the learned edited weights. However, we believe that the scope of the parameter updates in MEND are not localized in relation to the knowledge recall computation. Specifically, for BART/T5, their algorithms updated the MLP layers of the last 2 encoder & decoder blocks, and for GPT/BERT models, they edited the MLPs in the last 3 blocks.

Direct model editing methods attribute knowledge to particular groups of neurons in the model and selectively edit their activation to update knowledge. For instance, Dai et al. (2022) view MLPs in transformer architectures as key-value memories, and neurons in MLPs can be identified as knowledge neurons. They then edited the top four knowledge neurons involved in each relational fact to update knowledge. However, Meng et al. (2022a) believe that linear layers are forms of associative memory and proposed ROME. They view the dependencies between hidden variables in an autoregressive transformer model as a causal graph that contains many paths from the input to the output, and hypothesize a mechanism for the storage of factual associations where the middle layer MLP outputs accumulate information, which is copied to the last token by attention at higher layers. The weights in the middle layers are then rank-one updated by solving a constrained least-squares problem. Inspired by this, Meng et al. (2022b) created Mass-Editing Memory in a Transformer (MEMIT). Specifically, the hidden representations are changed by maximizing the model's prediction of the desired output y_e. Their method has been shown to be promising in scalability, specificity, and generalization. We will build upon their work by using causal mediation analysis to identify critical MLP layers in autoregressive transformer models.

3 Methods and Models

3.1 Knowledge Neurons

Based on well-established integrated gradient attribution method (Sundararajan et al., 2017) and key-value view (Geva et al., 2021), Dai et al. (2022) proposed Knowledge Neurons (KN) attribution method to identify knowledge neurons in the MLP key layers that are responsible for expressing factual knowledge. The identification method is to evaluate the contribution of each individual neuron to the knowledge prediction.

Knowledge Attribution Method To quantify the importance of a certain neuron, Dai et al. (2022) also introduced the knowledge attribution method,

$$\mathrm{Attr}\left(n_i^{(l)}\right) = \bar{n}_i^{(l)} \int_{\alpha=0}^{1} \frac{\partial \mathrm{P}_x\left(\alpha \bar{n}_i^{(l)}\right)}{\partial n_i^{(l)}} d\alpha,$$

where $n_i^{(l)} = \hat{n}_i^{(l)}$ denotes the i-th intermediate neuron in the l-th MLP key layer with its assigned value $\hat{n}_i^{(l)}$; $\bar{n}_i^{(l)}$ stands for the original value of the neuron $n_i^{(l)}$; $\mathrm{P}_x\left(\hat{n}_i^{(l)}\right) = p\left(y^* \mid x, n_i^{(l)} = \hat{n}_i^{(l)}\right)$ denotes the probability of the model predicting the correct answer y^* given an input x and the neuron value assignment; $\frac{\partial \mathrm{P}_x\left(\alpha \bar{n}_i^{(l)}\right)}{\partial n_i^{(l)}} d\alpha$ denotes the gradient of the output with respect to neuron $n_i^{(l)}$. Intuitively, if a neuron has a high impact on the output prediction, its calculated attribution score would be high. However, the integration presented above is often computationally intractable, so we followed the Riemann integration approximation used by the authors, $\tilde{\mathrm{Attr}}\left(n_i^{(l)}\right) = \frac{\bar{n}_i^{(l)}}{m} \sum_{k=1}^{m} \frac{\partial \mathrm{P}_x\left(\frac{k}{m} \bar{n}_i^{(l)}\right)}{\partial n_i^{(l)}}$ with the same approximation steps, $m = 20$.

Refined Knowledge Neurons Using the knowledge attribution method, we could identify a coarse set of neurons for a single prompt. However, the identified neurons may not all be responsible for expressing factual knowledge, they could encode other linguistic phenomena as well. To filter out these 'false-positive' neurons not related to expressing knowledge, Dai et al. (2022) also proposed a refinement strategy. First, sample n diverse prompts that express the same knowledge; Then, for each prompt, calculate the attribution score of neurons and keep the neurons in the coarse set with attribution scores greater than a given threshold t; Lastly, considering all the coarse sets of all the n prompts, we preserve the knowledge neurons that are identified as knowledge neurons in more than $p\%$ prompts.

3.2 In-Context Learning

If the GPT model is given a bare prompt, for example, we ask it to predict *"The capital of France is _"*, it will assign a very low probability to the word *Paris* for the next token prediction. This is be-

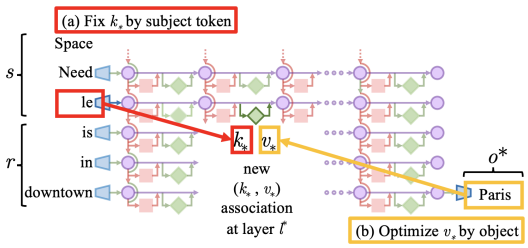


Figure 2: **Editing one MLP layer with ROME**(Meng et al., 2022a). To update the model on the fact that Space Needle is located in Paris, ROME inserts a new (k_*, v_*) pair into layer l^*

cause many legit sentences can stem from the given prompt, and the model does not know which one to choose. However, low prediction output of the correct token will bring about more noise from random neurons to the identification of the true knowledge neurons. Inspired by Brown et al. (2020), we injected context into the data and successfully increased correct token predicting probability by 50%. For each prompt template, we create the context by filling the prompt template with sampled⟨s, o⟩pairs.

3.3 Causal Mediation Analysis

Given an autoregressive transformer model f_θ, to calculate the contribution of an intermediate variable in causal graphs, we can observe all of the internal activations in f_θ in 3 different runs. In the **clean run**, a factual prompt x is passed into the f_θ, and its hidden states, $\{h_i^l | i \in [1,T], l \in [1,L]\}$, are collected. In the **corrupted run**, the hidden states of the embeddings are corrupted by adding Gaussian noise: $h_i^0 = h_i^0 + \epsilon$, where $\epsilon \sim \mathcal{N}(0;\nu)$. In the **restoration run**, each token embedding is restored to its noiseless state, and the state that recovers the prediction ability is intuitively the most causally important. Let o denote the subject of the prompt x, and $P(o), P_*(o), P_{*,cleanh_i^l}(o)$ denote the probability of predicting o under the clean, corrupted, and restored runs respectively. The indirect effect IE is defined as: $IE = P_{*,cleanh_i^l}(o) - P_*(o)$. We define the causal neuron for a prompt as the MLP hidden state that generates the most indirect effect: $\arg\max_{i,l} P_{*,cleanh_i^l}(o) - P_*(o)$

3.4 Knowledge Editing

While we did not propose a novel editing model, the new pipeline proposed here focuses on the analysis of knowledge localization. Specifically, for each knowledge tuple and prompt to edit in the *counterfact* dataset, we first identify the causal neuron as defined in the causal mediation analysis section, then we perform the editing method **KN-E** (Knowledge Neuron Editing) proposed by Dai et al. (2021): $h_i^l = h_i^l - \lambda_1 t + \lambda_2 t'$, where t denotes the embedding of o^c, and t' denotes the embedding of o^* (the object of the edited knowledge). We call this pipeline **CNE** (Causal Neuron Editing). We compare CNE to the original editing method defined by Dai et al. (2021), and **ROME** (Rank-One Model Editing) by Meng et al. (2022a). Note that our method differs from KN-E in the neuron identification step; KN-E uses knowledge attribution, but we use causal mediation analysis. Similar to KN-E, **ROME** edits the knowledge by inserting a new key-value pair into the MLP layers (Figure 2). However, ROME's editing step assumes that the key vector is always the midlayer MLP embedding of the subject's last token. For instance, in Figure 2, the key vector is generated from the embedding vector of the token le. Their updating step for MLP projection matrix W is defined as the following: for new key-value pair (k_*, v_*), minimize $||\tilde{W}K - V||$ such that $\tilde{W}k_* = v_*$ by setting

$$\tilde{W} = W + \Lambda(C^{-1}k_*)$$

where $C = KK^T$ is a constant that estimates the uncentered covariance of k, and $\Lambda = \frac{(v_* - Wk_*)}{(C^{-1}k_*)^T k_*}$.

4 Dataset and Evaluation

4.1 Datasets

context-PARAREL Following Dai et al. (2022), we tested the factual Knowledge Neurons through the fill-in-the-blank task using PARAREL dataset (Elazar et al., 2021b). PARAREL contains diverse prompt templates for 38 relational patterns from T-REx dataset (Elsahar et al., 2018). The dataset has high syntactic and lexical variations, which are necessary to filter out false-positive neurons to identify the true knowledge neurons. For every relational fact, we fill in the subject entity in the template and make the model predict the object entity. However, since GPT is autoregressive in nature, which limits itself only to predicting the next token given the prompt, we filter out prompt templates whose object entity is not located at the end of the sentence. To keep the diversity of prompt templates, we then filter out relations with fewer than four prompt templates. After that, for every remaining relation, we

Relations	Template	Example
P39(position)	[X] has the position of [Y]	**Cadoc** has the position of **bishop**.· · · John XXI has the position of [pope]
P108(employment)	[X] works for [Y]	**Terry Myerson** works for **Microsoft**.· · · Steve Jobs, who works for [Apple]

Table 1: Example prompts from context-PARAREL. [X] and [Y] are empty placeholders for the subject and object entities. Bold texts are randomly sampled subject and object pairs from the relation vocabulary to facilitate In-context Learning. Target words are inside the square brackets for prediction.

sample at most 100 $\langle s, o \rangle$ pairs per relation as subjects and objects as we do not have the computing resource to run over the whole dataset. Finally, we random sample 4 unique $\langle s, o \rangle$ pairs from the prompt template vocabulary and fill into the duplicate prompt templates to enable In-context learning (Section 3.2). After all these operations, we have 22 relations and 17042 prompts in total. We name this subset as *context-PARAREL*. We show some prompt examples in Table 1.

counterfact To evaluate the reliability, generality, and locality of model updates, we use the *counterfact* dataset proposed in Meng et al. (2022a). Each record in this dataset corresponds to an entry in the *context-PARAREL* dataset. A record contains knowledge tuples $t^c = (s, r, o^c)$, where r denotes the relation, s denotes the subject, and o^c denotes the counterfactual object related to the original factual object o. The goal of the dataset is for a model to edit factual objects into counterfactual objects from prompts. Besides paraphrasing and neighborhood prompts that test generalization and locality respectively, the dataset also contains generation prompts that demand deeper generalization from model editing. The dataset contains an overall 21,919 records with 645 relations. While the authors claimed that the generation prompts are hand-curated for each relation from *WikiData*, we found that the generation prompts of many records contain duplicates, which were removed during evaluation. This dataset is also used in casual mediation analysis during knowledge editing to locate the "casual neuron".

4.2 Evaluation Metric

Model Knolwedge Evaluation Due to the parameterized nature of the transformer models, we cannot directly know if it possesses a piece of knowledge. Following Petroni et al. (2019), we also evaluate if a model has factual knowledge by measuring its ability to predict the correct answer. The measurement is carried out using the fill-in-the-blank task. Specifically, succeeding Petroni et al. (2019), we encoded the task into relational fact prediction. The relational fact is expressed in a subject-relation-object triple $< s, r, o >$, where s is the subject entity, o is the object entity, and r is the relation between them. For example, given a relational fact, $\langle$ Phil Knight, Employment, Nike $\rangle$, a prompt could be *Phil Knight, who works for* _. The model is considered to grasp the knowledge if the model can successfully predict *Nike* with the highest probability.

Knowledge Editing Evaluation The metrics used in knowledge editing are similar to that of Meng et al. (2022a). Efficacy Score (ES) evaluates editing success and is the proportion of cases for which the predicted probability of the edited model on the new object is greater than that of the old object ($P(o^*) > P(o^c)$). Efficacy Magnitude (EM) is the main difference $P(o^*) - P(o^c)$. Paraphrase Success (PS) and Paraphrase Magnitude (PM) are generalization measures defined similarly, except the model is prompted with re-phrasings of the original statement. For testing specificity, Neighbourhood Success (NS) and Neighbourhood Magnitude (NM) are defined similarly, but we check the probability the model assigns to the old object ($P(o^c) > P(o^*)$ and $P(o^c) - P(o^*)$), given prompts about distinct but semantically-related subjects. The overall success (S) is then defined as the harmonic mean of ES, PS, and NS.

5 Experiments, Results, and Discussion

5.1 Experiment Setup*

We first conducted our experiments on BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019), which is one of the most

*The code for the project are available at https://github.com/SedimentaryRockStar/Localization_Model_Knowledge

popular bidirectional models in recent years. The model has 110M parameters and contains 12 transformer encoder blocks. It has a hidden size of 768 and MLP intermediate hidden size of 3072. We then focused on GPT-2 (Generative Pre-trained Transformer) (Radford et al., 2019), an autoregressive model with 124M parameters. The GPT-2 model has 12 transformer decoder blocks with a hidden size of 768. Its MLP layer is represented by the 1×1 Convolutional Neural Network with an intermediate hidden size of 3072. The models are based on Huggingface pre-trained Transformer implementations (Wolf et al., 2020) with PyTorch (Paszke et al., 2019) as the backend. For the experiments involving Knowledge Neurons algorithms, we set the adaptive threshold t to 0.3 times the highest attribution score, sharing threshold $p\%$ to 0.5. All the experiments are run on Nvidia Tesla P100 GPUs.

For causal mediation analysis, the hyperparameters are the following: sample $= 10$, window $= 10, \nu = 0.1$. For knowledge editing, the hyperparameters are the following: $p = 0.5, \lambda_1 = 1, \lambda_2 = 8, lr = 1$. For ROME, the hyperparameters are the following: for GPT-2, the layer we edit is 5 (derived from averaged indirect effect), and for GPT-2 medium, the layer we edit is 8. Due to limited time and computational resources, for both models, instead of sweeping, we used the default hyperparameters for the number of samples (10000), learning rate (0.5), KL factor (0.0625), etc. We also only evaluated model editing on 1,000 records in the counterfact dataset.

5.2 Knowledge Neurons in GPT

We extend the original Knowledge Neuron work from BERT to GPT and then examine its effect on knowledge expression in the GPT model as shown in Figure 3 and 4. Given a relational fact prompt, we followed Dai et al. (2022) to manipulate the knowledge neurons in two ways: (1) Suppressing: set the activation value of the knowledge neurons to 0; (2) Enhancing: double the activation value of the knowledge neurons. For every relation, we plot the average change difference of the prediction probability for the correct answer. More than that, to investigate the manipulation effect on other unrelated knowledge, we also sample random unrelated facts and plot their changes in probability difference due to the manipulation.

As shown in Figure 3, suppressing identified knowledge neurons on GPT-2 did not lead to a prominent decrease in all relations (0.93% on average); there are only spikes of changes on certain relations. Among all the relations, P279(type) has the highest change in the probability for the correct answer (10.15%), while P740(organization) has the smallest change (6.91e-14%).

In Figure 4, we observe similar results for enhancing the identified knowledge neurons on GPT-2, with P279 having the highest change of 6.06% and P27(citizenship) having the smallest change of 4.80e-15%.

Seen from the plots, the manipulation of knowledge neurons seems not to impact other unrelated knowledge very much; we only see a decrease in the unrelated facts for the correct probability by 0.08% when suppressing the knowledge neurons and an increase of 0.12% when enhancing the knowledge neurons.

To summarize, the manipulation of knowledge neurons in GPT-2 did not have a comparable effect as in BERT. Based on our experiments on BERT with the same experiment setting, we observe a decrease of 5.94% when suppressing and an increase of 4.86% when enhancing.

5.3 Knowledge Neuron distributions

According to the results from Dai et al. (2022) and our experiments, the knowledge neurons are distributed mostly in the last few layers of MLP in BERT. However, Figure 5 shows a different distribution pattern for the GPT model. The first layer of MLP, together with its lower layers captures a rather high proportion of knowledge neurons.

5.4 Word Embedding Perturbation Test

To further investigate the disagreement in the distribution behavior of the knowledge neurons, we perform the word embedding perturbation test on both models. Concretely, for every input prompt x, it could be embedded as a sequence of $[e_0, e_1, \cdots, e_l]$, we inject Gaussian noise by setting $e_i := e_i + \epsilon, \forall i \in \{0, \cdots, l\}$, where l is the prompt length and $\epsilon \sim \mathcal{N}(0; \nu)$. Meng et al. (2022a) argued that the noise must be large enough to create a noticeable effect. We then set $\nu = \frac{1}{3}\lfloor mean(abs((E)) \rfloor$, where E is the model embedding matrix. Though the change in the embedding is subtle, we observe a dramatic decrease in the correct answer prediction in GPT-2 (9.51%), compared to the decrease in BERT (0.51%). The results indicate that GPT-2

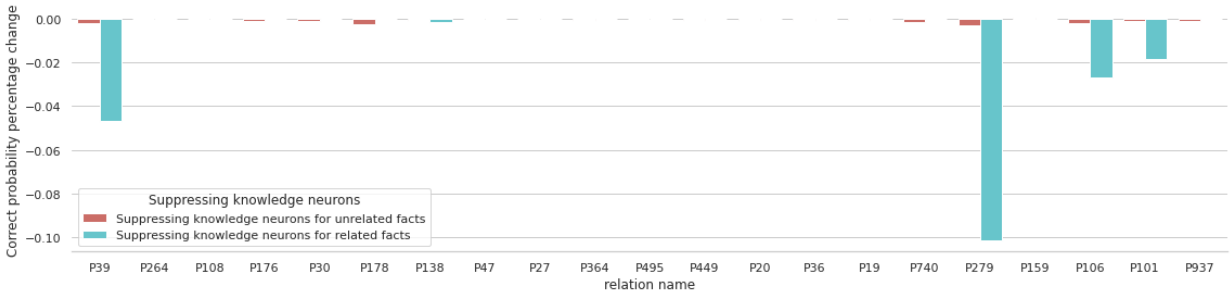


Figure 3: Results of knowledge neuron suppression on GPT-2. Suppressing knowledge neurons decreases correct token prediction probability by 0.93% on average for related facts over all the relations, 0.08% for unrelated facts

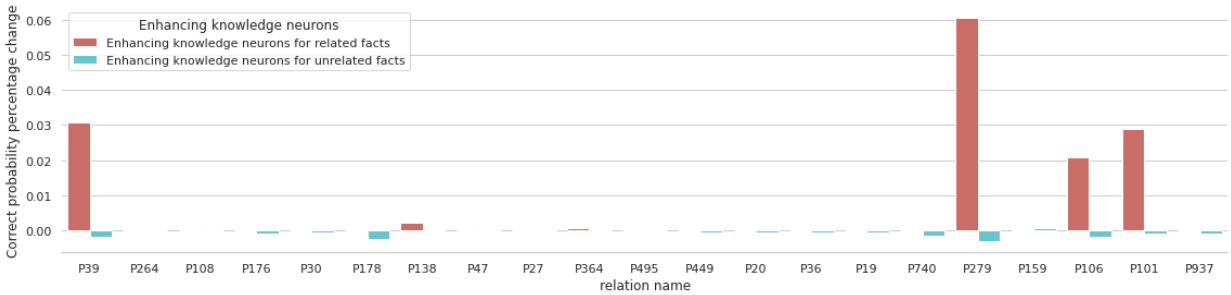
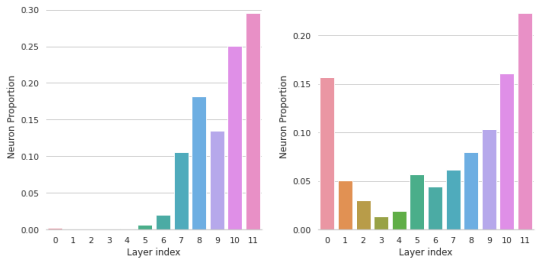


Figure 4: Results of knowledge neuron enhancement on GPT-2. Enhancing knowledge neurons increases correct token prediction probability by 0.48% on average for related facts over all thee relations, 0.12% for unrelated facts



(a) BERT (b) GPT-2

Figure 5: Knowledge Neuron Distribution for BERT and GPT-2.

highly relies on the prompt to make the correct prediction. This experiment provides insight into the spike of knowledge neuron distribution in GPT-2. We hypothesize that the neurons in the first layers of GPT-2 play a significant role in processing the prompt information passed through word embedding. A tiny change in the input (word embedding this time) would cause a big shift in the final prediction.

5.5 Syntactic Knowledge Neurons

According to Dai et al. (2022), knowledge neurons are limited to expressing factual knowledge as diversified prompts would filter out 'false-positive' neurons responsible for syntactic knowledge. We challenge this idea by identifying Syntactic Knowledge Neurons. To directly assess the syntactic knowledge of the model is difficult, we then took an indirect path by identifying the Knowledge Neurons for punctuation marks.

English Dataset for Punctuation Prediction: We examine our hypothesis using English Dataset for Punctuation Prediction (Rahin, 2021). This dataset is curated using the online OPUS corpus of the English language with pre-cleaning and pre-formatting procedures conducted by the authors. The dataset contains 1031725 sentences ending with various kinds of punctuation marks, including 685608 sentences ending in '.', 6770 sentences ending with ',', 56821 sentences ending with '!', 119442 sentences ending with '?', 1166 sentences ending with ';'. We only performed our experiments on the major end-of-sentence punctuation marks listed above. To address the problem of imbalanced sentence distribution of these punctuation marks, we sampled 200 sentences from each punctuation category to make a fair identification. We perform the task by making the model predict the next token given a punctuation mark truncated prompt. And then, we use the knowledge attribution algorithm to identify the knowledge neurons

Model \ Punctuation	'.'	','	'!'	'?'	';'
BERT-base-uncased	$n_9^{(927)}(90\%)$	$n_{11}^{(1205)}(90\%)$	$n_{10}^{(431)}, n_{11}^{(240)}, n_{11}^{(2399)}(60\%)$	$n_9^{(1673)}(80\%)$	$n_{11}^{(240)}(80\%)$
GPT-2	$n_9^{(831)}(90\%)$	$n_9^{(831)}(90\%)$	$n_9^{(831)}(90\%)$	$n_9^{(831)}(80\%)$	$n_9^{(831)}(90\%)$

Table 2: Punctuation Mark Knowledge Neurons. $n_i^{(j)}$ stands for the j-th neuron in the i-th MLP layer. $(p\%)$ means the neurons are identified in more than $p\%$ prompts' coarse sets.

with the same hyperparameter setting for t.

Results: As shown in Table 2, we have successfully identified a set of knowledge neurons that are dedicated to the syntactic punctuation mark knowledge. This confirms our hypothesis that knowledge neurons are not limited to expressing factual knowledge. We found that these syntactic knowledge neurons are all located in the upper layers of the model with a relatively high sharing percentage. This might be a potential implication that the transformer model would store its syntactic knowledge in the upper-layer MLPs. Surprisingly, GPT-2 model has one identified knowledge neuron that is shared by all the categories. This implies that GPT-2 model may have a highly concentrated storage of its syntactic knowledge.

5.6 Knowledge Editing

Results As shown in Table 3, ROME is surprisingly effective at editing smaller language models. It is able to achieve almost perfect rewrite success and high rewrite magnitude for both GPT-2 and GPT-2 medium. For GPT-2, editing the second last layer is not significantly different from editing the causal layer in terms of efficacy, generalization, and specificity, but this is not true for GPT-2 medium. It is also worth noting this although the average indirect effect of GPT-2 is the highest in the later MLP layers (Figure 6), editing the 5^{th} layer showed better results. Editing knowledge with KNE is also surprisingly effective for GPT-2, but not for GPT-2 medium. It is also evident that ROME achieved better generalization and specificity compared to KN. We also found that editing with CN was not effective at all. The model emitted the same probability before and after the edits.

Discussion and Analysis Since our editor CN was not editing the knowledge in the models, it was not able to give us further insights into which MLP layers are more important for GPT-2 and GPT-2 medium. We suspect that this is because, during editing, we only selected the most important causal neuron. Indeed, in cases where KNE found less than 5 refined neurons, the edits were also not effective. Intuitively, since KNE and CNE add and subtract word embeddings, applying these operations to only one neuron would have little to no effect. However, it could also mean that editing the causal neurons is not effective. We also found that the base prediction probability of GPT-2 was very low. In other words, it was not able to answer the prompts correctly. This sometimes caused the traces of the states in restored runs to be indistinguishable from each other (see Figure 7 in Appendix), which would hinder the ability of CNE to locate the causal neurons. However, the results from ROME Last are intriguing. While the average trace indicated that the last MLP layers of GPT-2 are the most important, editing the middle layers with ROME provided slightly better results than editing the last layers. Note that this was not true for GPT-2 medium. We suspect that this is because the knowledge storage in smaller language models is very concentrated, and the MLP layers are all intertwined. Another explanation for this is that ROME is acting as a general fine-tuning editor that maximizes the probability of prediction for the new fact. Since the prediction probability was already low in GPT-2, there were no key-value associations to edit in the first place, and ROME was fine-tuning the model to answer the prompts. It would be interesting to examine ROME's general ability to increase $P(o^c)$. We also believe that the evaluation metric proposed by Meng et al. (2022a) is not comprehensive. For instance, the overall score is the harmonic mean of ES, PS, and NS. This caused KN GPT-2 medium and GPT-2 to have similar scores. However, KN applied on GPT-2 medium was not effective as its magnitude was on average low (-0.3). While they also claimed that the edits are fast, we found that the pre-caching step takes a considerable amount of time (13063 seconds for GPT-2 medium). This means that, on average, KN applied on GPT-2 medium only marginally increased

Editor	Score	Efficacy		Generalization		Specificity	
	S	ES	EM	PS	PM	NS	NM
GPT-2	36.1	28.1 (44.9)	-1.7 (5.9)	30.1 (39.7)	-1.8 (5.4)	70.8 (32.6)	1.9 (4.6)
ROME	80.8	100.0 (0.0)	98.6 (3.9)	98.6 (9.7)	71.6 (32.3)	63.9 (34.1)	0.3 (8.1)
ROME Last	73.8	99.7 (5.5)	94.4 (14.3)	95.5 (17.9)	69.8 (33.5)	49.6 (34.9)	-4.8 (14.8)
KN	50.5	70.7 (45.5)	3.6 (12.3)	66.0 (43.6)	3.8 (12.4)	33.2 (38.6)	-3.4 (10.8)
CN	36.1	28.1 (44.9)	-1.7 (5.1)	30.1 (39.7)	-1.8 (5.1)	71.8 (31.6)	1.8 (3.6)
GPT-2 Medium	33.4	25.0 (1.0)	-3.3 (0.2)	27.4 (0.9)	-3.0 (0.2)	74.9 (0.7)	3.6 (0.2)
ROME	87.4	100.0 (0.0)	94.9 (0.3)	68.5 (0.9)	6.1 (0.4)	51.3 (0.8)	-1.7 (0.3)
ROME Last	33.4	25.1 (1.1)	-3.4 (9.9)	26.3 (38.1)	-2.9 (7.7)	75.5 (29.9)	3.9 (7.5)
KN	50.3	51.7 (50.0)	-0.3 (6.1)	47.3 (44.9)	-0.7 (6.4)	52.1 (39.2)	0.8 (6.1)

Table 3: **Quantitative Editing Results**. 95% confidence intervals are in parentheses. The rows GPT-2 and GPT-2 medium indicate the scores of the original model (before being edited). "ROME Last" indicates applying ROME on the second last layer of the corresponding GPT-2 model (10^{th} layer for GPT-2, 22^{th} layer for GPT-2 medium). CN corresponds to Causal Neuron Editing, and KN denotes Knowledge Neuron Editing.

$P(o^*)$. Finally, it is also worth noting that for GPT-2, knowledge neuron and causal tracing agree on the fact that the later MLP layers are the most important.

5.7 Limitations

Due to the teaching GPUs being down for a long time, we cannot run the datasets at full scale, and we could not deploy larger models to expand the dimensions of our experiments (for instance CN-E on GPT-2 medium).

6 Conclusion

As the scale of a language model increases, its performance increases (Brown et al., 2020). However, in this work, we decided to decipher where knowledge is stored in smaller language models. To do so, we applied two techniques: Knowledge Neuron and Causal Mediation Analysis. We found that in GPT-2, most of the knowledge is indeed stored in the upper MLP layers, which also contain one neuron that is responsible for specific syntactic knowledge. We then further analyzed the storage of knowledge by editing the specific knowledge with reliability, generality, and locality. From this, we found that 2 model editors (KN and ROME) that mainly edited the last MLP layers of the model were effective. However, as the scale of the language model grows, this is not necessarily true. We suspect that as the number of MLP layers increases, some layers become redundant (Voita et al., 2019). As mentioned, due to time and computational resource constraints, more experiments need to be done to provide a comprehensive overview. Future work can focus on analyzing the effect of enhancing/suppressing knowledge neurons per MLP layer, applying CNE to the top-k causal neurons, creating a better metric for combining the metrics used, and evaluating knowledge editing for multiple edits.

7 Statement of Contributions

Steve Wen did the experiments on knowledge neurons related tasks and wrote their corresponding part of the report. Mark Bai conducted experiments on causal mediation analysis and knowledge editing experiments, and wrote the corresponding parts. We would like to thank Zichao Li for his support as he gave us directions and suggestions.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, and Furu Wei. 2021. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Pro-*

ceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021a. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021b. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

David Ha, Andrew M. Dai, and Quoc V. Le. 2017. Hypernetworks. In *International Conference on Learning Representations*.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. 2021. Self-attention attribution: Interpreting information interactions inside transformer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12963–12971.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021. Beliefbank: Adding memory to a pre-trained language model for a systematic notion of belief. *arXiv preprint arXiv:2109.14723*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022a. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2022b. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. 2022a. Fast model editing at scale. In *International Conference on Learning Representations*.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022b. Memory-based model editing at scale. In *International Conference on Machine Learning*, pages 15817–15831. PMLR.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Judea Pearl. 2022. Direct and indirect effects. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 373–392.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Md. Rezwan Shahrior Rahin. 2021. English dataset for punctuation prediction.

Matthew Sotoudeh and A Thakur. 2019. Correcting deep neural networks with small, generalizing patches. In *Workshop on safety and robustness in decision making*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in Neural Information Processing Systems*, 33:12388–12401.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*.

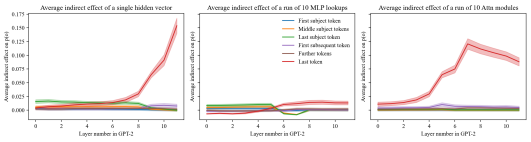


Figure 6: **Mean causal traces of GPT-2 over a sample of 1000 factual statements** Shown as a line plot with 95% confidence interval.

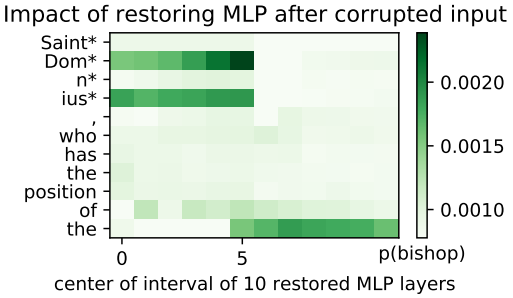


Figure 7: **Example of low prediction probability in causal traces for GPT-2**

A Appendix

A.1 Related Work

Memory augmentation approaches the problem in a non-parametric way; by "wrapping" the original classifiers with auxiliary memory modules (Mitchell et al., 2022b; Kassner et al., 2021). For instance, Semi-Parametric Editing with a Retrieval-Augmented Counterfactual Model (SERAC) stores knowledge edits an explicit memory and learns to reason over them to modulate the base model's predictions as needed It consists of three components: an explicit cache of knowledge edits, a counterfactual model that predicts in the output space $\mathcal{Y}$ based on the counterfactual world described by the explicit cache, and a scope classifier that directs post-edit inputs to the counterfactual model or the original model. SERAC displayed impressive performance on edit success for small classifiers; the reliability was high with low computational cost. However, this method relies on the same dataset of edits (similar to MEND) for training the auxiliary modules. Also, as the number of edits grows continuously, the explicit memory cache would grow out of bound, so it is not scalable in some settings.

A.2 Causal Tracing

Sample of causal tracing with low probability: Figure 7

A.3 Datasets

Sample of the counterfact dataset generation prompt: Figure 8

```
GPT-2 XL: Pierre Curie often
collaborated with his wife,
Marie Curie, on ... radiation
research
Insert Counterfactual: Pierre
Curie's area of work is
medicine
Correct behaviour: Pierre Curie's
expertise is in the field of
medicine and medicine in science.
Incorrect behaviour: Robert A.
Millikan's area of work is
medicine. His area of expertise
is the study of the immune system.
```

Figure 8: Example of counterfact generation prompt